Computer Science Department

TECHNICAL REPORT

THE VOLUME OF THE UNION OF MANY SPHERES
AND POINT INCLUSION PROBLEMS

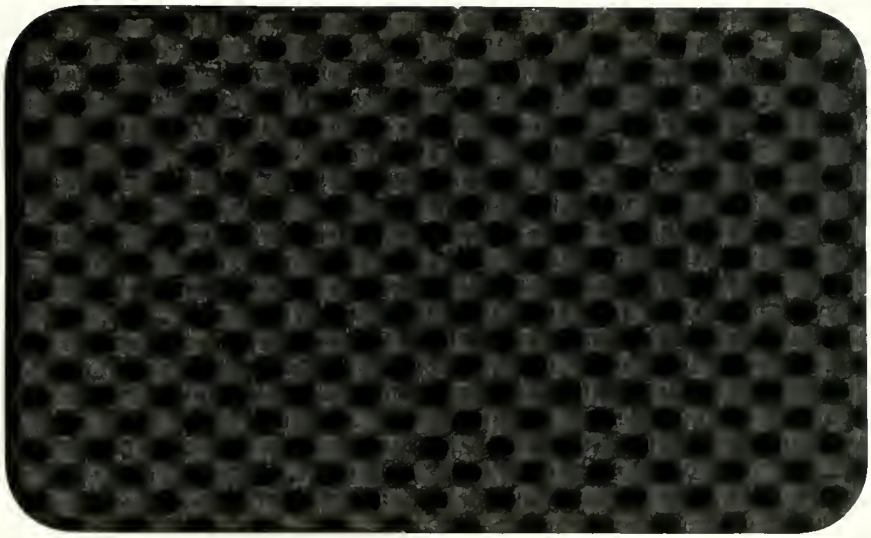
By

Paul G. Spirakis

August 1984
Report # 133

NEW YORK UNIVERSITY

Department of Computer Science
Courant Institute of Mathematical Sciences
251 MERCER STREET, NEW YORK, N.Y. 10012



THE VOLUME OF THE UNION OF MANY SPHERES
AND POINT INCLUSION PROBLEMS

By

Paul G. Spirakis

August 1984
Report # 133

c.1

Abstract

We present her an $O(n)$ probabilistic algorithm for computing the volume of the union of n spheres of possibly different radii. The methd, which is an application of techniques developed by [Karp, Luby, 83], can be extended, in a straightforward manner, to compute the volume of the union of n objects (where each of them has an easy description e.g. boxes of spheres) in k dimensions. Its time complexity is then $O(nk)$. We also examine the related problem of computing the number of spheres (or disks, in the plane) among a given set of spheres, containing a given point. For the case of n disks of the same radius r, we can answer such a query in time $O(\log^2 n$ and $O(n^3)$ preprocessing space.

For the more general problem of n spheres of different radii, we can answer such queries in $O(\log^2 n)$ time and storage $O(n \log n)$, following a technique of [Chazelle, 83]. This leads to an $O(n \sqrt{n})$ expected time union estimation algorithm.

The probabilistic estimation of the union follows ideas developed by R. Karp and M. Luby (see [Karp, Luby, 83]). Some of our notation is heavily affected by their notation.

We also show how to use the above methods to tet if n spheres have a (nonzero measure) intersection, in probabilistic time $O(n)$.

1. Introduction

The problem of estimating the area of the union of many circles in the plane was first posed by [Shamos, 78]. It is straightforward to solve this problem by an $O(n^2)$ time deterministic algorithm. We show here how to use Monte Carlo techniques (developed by [Karp, Luby, 83] for estimation of the failure probability of an n-component system) to get an $O(n)$ probabilistic algorithm. This algorithm can trivially be extended to compute the volume of the union of n spheres (or boxes, or any collection of n fixed description objects) in k dimensions, in time $O(nk)$. The accuracy of the algorithm is controlled by the algorithm implementer. Its running time is optimal in the sense that $O(n)$ time is needed to compute the sum of n real numbers, under quite general models of computation. An application of this algorithm is a probabilistic $O(n)$ time method to test if n given spheres have a nonzero measure intersection. This can also be extended to test for intersections in k dimensions. No efficient algorithms for the problem of computing volumes of unions of objects in more than two dimensions were presented in the past. The fastest up to now algorithm for testing if n spheres intersect was invented by [Schwartz, Hopcroft, Sharir, 83] and runs in time $O(n \log^2 n)$. This algorithm detects also zero measure intersections.

The time complexity of standard Monte Carlo techniques for estimating volumes of union of many objects depends crucially on time-efficient solutions of the <u>cardinality of point enclosure</u> problem i.e., given n objects and a point X, find how many of the objects contain X. Let us call this number the <u>cover</u> of X. Our $O(n)$ probabilistic volume of union estimation technique uses a method of estimating covers which can be very slow (up to $O(n)$) but has the nice property that the slower it is, the less is the number of random points whose cover has to be found. Since the n objects are given in advance and the covers of many points have to be estimated, one can use preprocessing ideas to solve the cardinality of point enclosure problems. For the case of n disks of the same radius, r, in the plane, we can solve the cover estimation problem in $O(\log^2 n)$ time, with $O(n^3)$ preprocessing space. This method however uses as a preprocessing idea the estimation of all k-Voronoi diagrams of the n centers for k = 1 up

to n and has preprocessing time $O(n^3 \log n)$ which is prohibitive for use of the method for union estimation. For the (more general) case of n spheres of different radii in 3 dimensions, we can answer such queries in $O(\text{cover}(X)+\log^2 n)$ time each, with preprocessing storage $O(n \log n)$ and also preprocessing time $O(n \log n)$. This leads to an $O(n \log^2 n)$ Monte Carlo technique for estimation of the area of the union of n spheres, in case the maximum cover is $O(\log^2 n)$. We show how to get an <u>expected</u> $O(n\sqrt{n})$ Monte Carlo technique, in case the maximum cover is $\Omega(\log^2 n)$.

Extended MC

2. Definitions and useful remarks

2.1 Notation

Let $C_1,\ldots,C_n$ be n spheres of radii $r_1,\ldots,r_n$ and center vector coordinates $\vec{v}_1,\ldots,\vec{v}_n$. Let U denote the volume of the union $\bigcup_{i=1}^{n} C_i$. Let $V(C_i)$ be the volume of the sphere C_i. Let Σ be the sum of the volumes of the n spheres. Let $\hat{U}$ be the estimate of U, produced by a randomzied algorithm MC (for Monte Carlo). Following [Karp, Luby, 83] we have:

<u>Definition:</u> MC is called an (ε,δ) algorithm of time f(n), if $\forall\, \varepsilon,\delta \in (0,1)$, one can select a running time $f(\varepsilon,\delta,n)$ of MC, such that the estimate $\hat{U}$ produced by MC will satisfy

$$\text{Prob}\left\{ \left| \frac{\hat{U}-U}{U} \right| > \varepsilon \right\} \leq \delta \qquad \text{(EQ1)}$$

<u>Definition:</u> If, for fixed ε, δ, $f(n) = O(n)$, then MC is called a <u>linear</u> (ε,δ) algorithm. In general, we call it a $g(n)$ (ε,δ) algorithm, if $f(n) = O(g(n))$ for fixed ε and δ.

2.2 Remarks for 2 dimensions

The part of the plane covered by the union of the n disks $C_1,\ldots,C_n$ is partitioned into nonoverlapping segments whose boundaries are circular arcs. It is easy to see that the number, m, of these segments is $O(n^2)$. Since the area U is the sum of the areas of these segments, one gets a straightforward $O(n^3)$ time and space algorithm for computing U, by keeping and modifying a list of the segments created by

the first i disks, for $i = 1$ to n. (Each segment is kept as a clockwise sequence of circular arcs surrounding the segment.) An $O(n^2)$ algorithm can be obtained by keeping just a list of circular arcs surrounding the current union, of the first i disks, for $i = 1$ to n. The $i+1^{th}$ disk will intersect the list in $O(i)$ point and one has to update the list by removing arcs which are interior to the new union and adding the pieces of the circumference of the $i+1^{th}$ disk which are not covered by the current union. This algorithm needs just $O(n)$ space and outputs the boundary of the union of the n disks. (The area can be found, from the boundary, in time $O(n)$, by the signed addition of $O(n)$ integrals, one for each arc of the boundary).

2.3 The Standard Monte Carlo approach.

Let's assume we can find (in time $a(n)$) a closed surface S (of volume V(S)) containing the n spheres. Assume also that we have a fast way (of time $\tau(n)$ per point selecton) of selecting points within S in a uniform random manner (i.e. the probability that a selected point falls into an elementary volume dv, around a specific pint P_0, is the same $\forall P_0$ in S plus the interior of S, and is equal to $\frac{1}{V(S)} \cdot dv$). Also, let $\beta(n)$ be the time it takes to decide if a given point P belongs to the union of the n spheres.

Standard MC

(1) Select $N \geq \frac{1}{\varepsilon^2\delta}\left(\frac{V(S)}{U} - 1\right)$ points, uniformly randomly. (Note that, to do this, we need to know just an upper bound B on V(S)/U).

(2) For each point selected in (1), test if it belongs to the union. Let $M \leq N$ be the number of points found to belong in the union.

(3) Output $\hat{U} = \frac{M}{N} \cdot V(S)$

Lemma 1. The (above) standard MC method is an (ε, δ) algorithm of time

$$O\left(a(n) + \tau(n)\beta(n)\,(B-1)\,\frac{1}{\varepsilon^2\delta} \right)$$

Proof: See full paper.

3. An $O(n)$ (ε,δ) algorithm

3.1 How to improve on the standard MC

A reasonable value of $a(n)$ is $O(n)$. (E.g. we can arbitrarily select a center, find the most distant of the centers of the C_i's to that center, add to the distance found the maximum radius and draw a sphere for S). If we don't use any preprocessing ideas, then $\beta(n) = O(n)$, leading to a time complexity of $O(n \cdot \tau(n) \cdot (\frac{V(S)}{U}))$. If we choose S to be the boundary of the union $\bigcup_{i=1}^{n} C_i$ itself, then $\frac{V(S)}{U} = 1$ but then the task of selecting points from the interior of S uniformly randomly becomes a hard task, since a point belonging to $k > 1$ spheres cannot be counted as a "whole" point. This leads to the following algorithm:

<u>Extended MC</u>

(1) Let S be the boundary of the union.

(2) Select N points ($N \geq cn/\varepsilon^2\delta$, c a constant) as follows: To select a random point in S, we select one C_j at random and then we select one point P_j, uniformly randomly within C_j.

(3) Compute $M = \sum_{j=1}^{N} (\text{cover}(P_j))^{-1}$

(4) Output M.

<u>Lemma 2</u>. Let $c(n)$ be the time to compute the cover of a point. Let $p(n)$ be the preprocessing time for this. Then, the extended MC is an (ε,δ) algorithm of time $O(n\, c(n) + p(n))$.

(Proof in the full paper). •

In Section 4 we show how to gext an extended MC algorithm of time $O(n \log^2 n)$, in cases where the maximum cover of any point is $O(\log^2 n)$, and an extended MC algorithm of <u>expected</u> time $O(n\sqrt{n})$ else.

3.2 A linear(ϵ,δ) algorithm.

We use here the following ideas: (a) We assign greater probability of selection to points belonging to spheres of bigger volume. (b) We estimate covers via a random test which takes a long time only if the cover is a small number. This means that the spheres do not overlap a lot and we exploit this knowledge in deciding how many points to select. The following technique is a modification of the [Karp, Luby, 83] techniques.

The linear MC (LMC) method

<u>BEGIN</u>

(1) Let c be a constant ≥ 20. Let initially F = 0, number of trials = 0, TIME = 0.

(2) Compute the sum Σ of the volumes of the spheres.

<u>STEP 1.</u> Randomly select sphere C_i with probability $p_i = \frac{V(C_i}{\Sigma}$

<u>STEP 2.</u> Randomly select a point $s \in C_i$

<u>STEP 3</u> $f_i = 0$

<u>repeat</u> <u>until</u> a C_j is selected, such that $s \in C_j$

<u>begin</u>

Randomly select C_j with probability $1/n$

$f := f+1$

test if $s \in C_j$

TIME := TIME + 1

<u>end</u>

$\hat{U} := \frac{f}{n} \cdot \Sigma$

<u>STEP 4</u> number of trials := number of trials + 1

$F := F + \hat{U}$

<u>if</u> TIME $< \frac{c \cdot n}{\delta\epsilon^2}$ <u>then</u> goto <u>STEP 1</u>

<u>STEP 5</u> Output $U^* = \frac{F}{\text{number.of.trials}}$

<u>END</u> (of algorithm MC).

Note that LMC stops in $O(\frac{n}{\delta\varepsilon^2} \cdot X(n))$ steps, where $x(n)$ = time for STEP 2 + time to test if $s \in C_j$.

For k dimensions, $x(n) = O(k)$. Hene, LMC runs in time $O(nk)$. Note thta LMC could be used for objects other than spheres, which have an $O(k)$ description. It remains to prove that LMC is an (ε, δ) algorithm.

3.3 Analysis of LMC

Definition: Let f_i be a random variable indicating the value of f at the end of the i^{th} trial of LMC.

Definition. Let $t(T)$ be the sum $f_1 + f_2 + \ldots + f_T$, i.e. the sum of f's of the first T trials.

Definition. Let $U^*(T)$ be the value of U^* computed if the algorithm makes T trials.

Definition. Let $\tilde{U}_i$ be the estimate of the area of the union for the i^{th} trial.

Definition. Let $T(t)$ be the number of trials done in TIME = t.

Definition. Let $U^*(t)$ be the output of LMC, given TIME = t.

Lemma 3. For every i:

(a) The mean value of f_i is $\frac{nU}{\Sigma}$

(b) The mean value of f_i^2 is $\leq \frac{2n^2U}{\Sigma}$

Proof: See full paper

•

Lemma 4. Let $\tilde{U}_i$ be the random variable indicating the value of $\tilde{U}$ in the i^{th} round. Then $\text{mean}(\tilde{U}_i) = U$ and

$$\text{mean}(\tilde{U}_i^2) \leq 2\,\Sigma \cdot U$$

Proof: See full paper.

•

Theorem 1. For any $\varepsilon, \delta \in (0,1)$

$$\text{Prob}\{|U^*(t) - U| > \varepsilon U\} \le \delta$$

where $t = \frac{c \cdot n}{\delta \varepsilon^2}$, c a constant. (See appendix for sketch of proof of Theorem 1.)

3.4 Application to intersection problems

Suppose we want to check if the volume of intersection of n spheres is at least $\alpha \cdot \Sigma$ where $\alpha \in (0,1)$ and Σ is the sum of their volumes. By intersection here we mean overlap of at least 2 spheres. Clearly, this is equivalent to $\Sigma - U \ge \alpha \Sigma$ i.e. $U \le \Sigma(1 - \alpha)$. Choose some $\varepsilon, \delta \in (0,1)$ and run LMC to get an estimate U^* of U. Check if $U^* \le \Sigma(1-\alpha)(1-\varepsilon)$. If yes, then with probability at least $1-\delta$ the area of intersection is at least $\alpha \cdot \Sigma$.

4. The cardinality of point enclosure.

4.1 The case of n disks of the same radius.

We are given n disks $C_1, \ldots, C_n$ of the same radius r. We want, given a point X in the plane, to find cover(X) efficiently. Clearly, cover(X) is the number of those of $C_1, \ldots, C_n$ which fall into a circle of center X and radius r. (This is like the disk retrieval query of [Cole,Yap,83] with a slight modification: Instead of the list of the centers falling into the circle, we want just their number). We first perturb the centers of $C_1, \ldots, C_n$ infinitessimally, so that each of them has only 1 k^{th} nearest neighbor (for $k = 1, \ldots, n$). (Similar techniques based on infinitessimal perturbations were used in [Schwartz,Sharir,83] to resolve degenerate configurations arising in other geometric problems). Then, we do the following preprocessing: We build the k^{th}-nearest neighbor diagram (each region in this diagram has the same k^{th} nearest neighbor) for $k = 1, \ldots, n$. It is easy to see that the k^{th}-nearest neighbor, NN_k, diagram is a coarsening of the superposition of the (k-1)-Voronoi and the k-Voronoi diagrams (see [Lee,82] for definitions and construction). The k-Voronoi diagram has size O(nk); if one examines the method of constructing the k-Voronoi diagram from the (k-1)-Voronoi diagram, he will notice that the edges of the two diagrams only intersect at their endpoints. So, the size of the k^{th}-nearest neighbor diagram is O(nk). This leads to a total

preprocessing time of $O(n^3 \log n)$ and total preprocessing storage of $O(n^3)$. To find cover(X), given X, we first search NN_{2^i} $(i = 1,2,\ldots)$ until a 2^k-nearest neighbor of X is found at distance > r. Then, we do a bounded binary search between $NN_{2^{k-1}}$ and NN_{2^k}. Let the furthest nearest neighbor N of X, for which distance(N,X) < r, be the j^{th}-nearest neighbor. Then, cover(X) = j. Clearly, the above procedure takes $O(\log^2 n)$ query time, per point X.

4.2 The case of n spheres in 3D

We use here a construction of [Chazelle,83]. We are given n spheres (of different radii, in general), in 3D and we want to compute cover(X) for points X. We first select an origin O and convert the spheres into boxes in polar coordinates r,ϕ,θ. These boxes are parallel to the axes r,ϕ,θ. We project the n boxes on one of the axes (say r) and organize the projections into a segment tree (see [Bentlye,80] and [Chazelle,83]. With this representation, each node v of the segment tree spans an interval I(v) given by the union of its leaves' intervals. Let w be the father of v. Let R(v) be the set of boxes whose projections cover I(v) but not I(w). We define now a "recursive" data structure $T_3(S)$ (where $S = \{C_1,\ldots,C_n\}$) consisting of the above segment ree, and for each v a pointer to the structure $T_2(V)$ where V is the projection of R(v) to the plane (ϕ,θ).

A query for cover(X) will now be answered by converting X to polar coordinates and searching for the r-coordinate in the segment tree, applying the algorithm recursively to each structure $T_2(v)$ encountered. One can easily show the query time to be $O(\text{cover}(X) + \log^2 n)$, with both storage and preprocessing time to be $O(n \log n)$. If it is guaranteed that for all X, $\text{cover}(X) \leq \log^2 n$, then the above procedure gives us an extended-MC (ε,δ) algorithm of time $O(n \log^2 n)$.

If, however, the spheres may overlap a lot, then the best we can do is stop the recursion as soon as it seems that $\text{cover}(X) \geq \sqrt{n}$, and run the probabilistic cover algorithm of LMC. Recall that the (expected) time of that test is $\frac{n}{\text{cover}(X)}$, leading to an $O(\sqrt{n})$ per query, and to an $O(n\sqrt{n})$ extended MC (ε,δ) algorithm.

Further Research

We pose as open the problem of getting a cover algorithm of preprocessing time $< \sqrt{n}$ and query cost $< \sqrt{n}$.

References

[Chazelle,83] "Filtering Search: A New Approach to Query-Answering," by B. Chazelle, 24th Annual Symposium on Foundations of Computer Science, Nov. 1983.

[Cole,Yap,83] "Geometric Retrieal Problems," by R. Cole and C. Yap, 24th Annual Symposium on Foundations of Computer Science, Nov. 1983.

[Feller,57], An introduction to probability theory and its applications, Vol. 1, Wiley, 1957.

[Karp,Luby,83] "Monte Carlo Algorithms for Enumeration and Reliability Problems" by R. Karp and M. Luby, 24th Annual Symposium on Foundations of Computer Science, Nov. 1983.

[Lee,82] "On k-Nearest Neighbor Voronoi Diagrams in the Plane" by D. Lee, IEEE Trans. Computing, No. 6, 1982.

[Schwartz, Sharir, 83] "On the Piano Mover's Problem: I. The Special Case of a Rigid Polygonal Body Moving Amidst Polygonal Barriers" by J.T. Schartz and M. Sharir in Comm. Pure Appl. Math., 1983.

[Sharir, 83] "Intersection and closest-pair problems for a set of planar objects" by M. Sharir, Courant Inst. Tech. Report No. 56, Feb. 1983.

Appendix

Proof sketch of Theorem 1 (see full paper for a complete proof): We employ here the technique of [Karp,Luby,83]. If T is the total number of trials, then

$$f_1 + \dots + f_T = t(T)$$

i.e.

$$T \cdot \text{mean}(f_i) = \text{mean}(t(T)) \, .$$

So, if $t = \frac{c_1 n}{\delta\varepsilon^2}$, for a constant c_1, we expect to do a number of trials

$T = \frac{c_1 n}{\delta\varepsilon^2} \cdot \frac{\Sigma}{U \cdot n} \leq \frac{c_1 n}{\delta\varepsilon^2}$, because $\frac{\Sigma}{U} \leq n$.

Let $k = \frac{c_1 \Sigma}{\delta\varepsilon^2 U}$ ("Average" number of trials).

If t" ($t'' = \frac{cn}{\delta\varepsilon^2}$, where $c = c_1(1+\beta)$) is the actual running time of the algorithm, let R(t") be the event

$$|U^*(t'') - U| > \varepsilon U$$

Clearly

$$\text{Prob}\{R(t'')\} = \text{Prob}\{R(t'') \text{ and } T(t'') < k\}$$

$$+ \text{Prob}\{R(t'') \text{ and } T(t'') \geq k\}$$

But

$$\text{Prob}\{R(t'') \text{ and } T(t'') < k\} \leq \text{Prob}\{T(t'') < k\}$$

$$\leq \text{Prob}\{f_1 + \dots + f_k > t''\}$$

$$\leq \frac{2\delta\varepsilon^2}{c_1\beta^2} \quad \text{(by Chebyshev inequality)}$$

Also

$$\text{Prob}\{R(t'') \text{ and } T(t'') \geq k\}$$

$$= \text{Prob}\{T(t'') \geq k \text{ and } \exists\, r \geq k: \left| \frac{\tilde{U}_1 + \dots + \tilde{U}_r}{r} - U \right| > \varepsilon U\}$$

$$\leq \text{Prob}\{\exists\, r \geq k: \left| \frac{\tilde{U}_1 + \dots + \tilde{U}_r}{r} - U \right| > \varepsilon U\}$$

Here one can use Kolmogorov's inequality (see e.g. [Feller,57]) to prove that

$$\mathrm{Prob}\{\ \exists r \geq k:\ \left|\frac{\tilde{U}_1+\ldots+\tilde{U}_r}{r} - U\right| > \varepsilon U\} \leq \frac{8\delta}{c_1}\ .$$

(See full paper.)

We conclude that

$$\mathrm{Prob}\{R(t'')\} < \frac{2\delta\varepsilon^2}{c_1\beta^2} + \frac{8\delta}{c_1}$$

Let $\delta' = \frac{\delta}{c_1}\left(\frac{2\varepsilon^2}{\beta^2} + 8\right)$. So, for any $\varepsilon, \delta' \in (0,1)$ we can choose $c = c_1(1+\beta)$ and β such that if $t'' = \frac{cn}{\delta'\varepsilon^2}$ then LMC is an (ε, δ') algorithm.

●